Creating a website with Razor Pages

Keywords:

"the middleware pipeline, which defines how an ASP.NET Core application responds to a request"

"Each piece of middleware can modify or handle an incoming request"

"your middleware pipeline will normally include the EndpointMiddleware"

"Razor Pages is responsible for generating the web pages that the user interacts with"

"serve as an API to backend server processes, to a mobile app, or to a client framework for building single-page applications (SPAs)"

"applications that serve both needs"

"creating server-side rendered HTML pages"

"MVC design pattern to see the benefits that can be achieved through its use"

"Razor Pages implements the MVC design pattern"

"add Razor Pages to an existing application"

"define page handlers to execute when your application receives a request"

"Web APIs still use the ASP.NET Core MVC framework"

"return data formatted for consumption in code"

"these frameworks are often responsible for handling all the business logic and UI code for your application"

"focus on the specific concepts being addressed"

Questions:

Q: What does the middleware pipeline define in an ASP.NET Core application?

A: The middleware pipeline defines how an ASP.NET Core application responds to a request.

Q: What can each piece of middleware do in the pipeline?

A: Each piece of middleware can modify or handle an incoming request before passing the request to the next middleware.

Q: What is typically included in the middleware pipeline of ASP.NET Core web applications?

A: The EndpointMiddleware is typically included in the middleware pipeline.

Q: What is responsible for generating the web pages that the user interacts with in an HTML web application?

A: Razor Pages is responsible for generating the web pages that the user interacts with.

Q: What formats does an ASP.NET Core API serve data in?

A: It serves data in machine-readable formats such as JSON or XML.

Q: What kind of application can serve both HTML and API responses?

A: Applications that serve both needs.

Q: What will you learn about in this chapter related to Razor Pages?

A: You'll learn how ASP.NET Core uses Razor Pages to handle creating server-side rendered HTML pages.

Q: Why has the MVC design pattern been adopted by many web frameworks?

A: Because of the benefits that can be achieved through its use.

Q: What does Razor Pages build on top of?

A: Razor Pages builds on top of the ASP.NET Core MVC framework.

Q: What can you define in Razor Pages to handle incoming requests?

A: You can define page handlers to execute when your application receives a request.

Q: What do Web APIs return instead of web pages?

A: They return data formatted for consumption in code.

Q: What are the next five chapters focused on?

A: The next five chapters all deal with a different aspect of the MVC pattern that makes up the MVC and Razor Pages frameworks.

Q: What should readers focus on if the framework behavior seems confusing?

A: Focus on the specific concepts being addressed.

An introduction to Razor Pages

Keywords:

"Razor Pages programming model"

"introduced in ASP.NET Core 2.0"

"build server-side rendered 'page-based' websites"

"using conventions where possible to reduce the amount of boilerplate code and configuration required"

"a page-based website is one in which the user browses between multiple pages"

"enters data into forms"

"applications like games or single-page applications (SPAs)"

"heavily interactive on the client side"

"a slightly more complex Razor Page"

"the MVC design pattern and how it applies to Razor Pages"

"how to add Razor Pages to your application"

"understanding of the overall design behind Razor Pages"

Questions:

Q: When was the Razor Pages programming model introduced?

A: The Razor Pages programming model was introduced in ASP.NET Core 2.0.

Q: What type of websites does the Razor Pages programming model help build?

A: It helps build server-side rendered “page-based” websites.

Q: What does Razor Pages use to reduce the amount of boilerplate code and configuration?

A: It uses conventions where possible.

Q: What defines a page-based website according to the passage?

A: A page-based website is one in which the user browses between multiple pages, enters data into forms, and generally consumes content.

Q: What kind of applications are described as heavily interactive on the client side?

A: Applications like games or single-page applications (SPAs).

Q: What kind of Razor Page example is examined in this section?

A: A slightly more complex Razor Page.

Q: What design pattern is discussed in relation to Razor Pages?

A: The MVC design pattern and how it applies to Razor Pages.

Q: What will you learn to add to your application in this section?

A: How to add Razor Pages to your application.

Q: What should you understand at the end of this section?

A: The overall design behind Razor Pages and how they relate to the MVC pattern.

Exploring a typical Razor Page

Keywords:

"very simple Razor Page"

"content-heavy marketing website"

"contain some logic, load data from a database, or use forms"

"slightly more complex Razor Page"

"to-do list application"

"display all the to-do items for a given category"

"PageModel code-behind for the Razor Page"

"page handler, OnGet"

"automatically populated by the Razor Page infrastructure"

"model binding"

"ToDoService"

"injected as a constructor argument using dependency injection"

"returns Page() at the end of the method"

"CategoryModel instance"

"access the Items property"

"central controller for the Razor Page"

"Model-View-Controller (MVC) design pattern"

"frameworks such as Django, Rails, and Spring MVC"

"mobile apps to rich-client desktop applications"

Questions:

Q: What did the simple Razor Page in chapter 2 do?

A: It didn’t contain any logic and instead just rendered the associated Razor view.

Q: What are common functions of more typical Razor Pages?

A: Contain some logic, load data from a database, or use forms to allow users to submit information.

Q: What application is the slightly more complex Razor Page taken from?

A: A to-do list application.

Q: What does the PageModel code-behind demonstrate compared to the basic example?

A: A variety of features compared to the basic example from chapter 2.

Q: What parameter does the OnGet page handler accept?

A: The OnGet page handler accepts a method parameter, category.

Q: How is the category parameter populated?

A: It is automatically populated by the Razor Page infrastructure using values from the incoming request in a process called model binding.

Q: What service does the handler interact with using dependency injection?

A: It interacts with the ToDoService, which is injected as a constructor argument.

Q: What does the handler return at the end of the method?

A: The handler returns Page() at the end of the method.

Q: What does the Razor View use to build the HTML sent to the user?

A: It uses the Items property that is set by the handler.

Q: What does the pattern of interactions in the Razor Page resemble?

A: It looks like the Model-View-Controller (MVC) design pattern.

Q: What frameworks use the MVC pattern in web development?

A: Frameworks such as Django, Rails, and Spring MVC.

Q: Where else can the MVC pattern be found beyond web frameworks?

A: In everything from mobile apps to rich-client desktop applications.

The MVC design pattern

Keywords:

"MVC design pattern"

"rich-client graphical user interface (GUI) apps"

"separate the management and manipulation of data from its visual representation"

"Razor Page handles different aspects of a request"

"Model—This is the data that needs to be displayed"

"View—The template that displays the data"

"Controller—This updates the model and provides the data"

"page handler in Razor Pages"

"Each component in the MVC design pattern is responsible for a single aspect"

"the order of events when an application responds to a user interaction"

"controller (the Razor Page handler) receives the request"

"controller either fetches the requested data or it updates the data"

"view uses the data contained in the model to generate the UI"

"controller serves as the entry point for the interaction"

"controller handles it"

"the model is considered to be a complex beast"

"Razor Page PageModel class is not the model we're talking about"

"Razor Page handler as a mini controller"

"the model being independent of the view"

"the view is only responsible for generating the final representation of the data"

"MVC design pattern in relation to web applications"

Questions:

Q: What is the main purpose of the MVC design pattern?

A: To separate the management and manipulation of data from its visual representation.

Q: What kind of apps was the original MVC pattern designed for?

A: Rich-client graphical user interface (GUI) apps.

Q: What are the three components that make up the MVC design pattern?

A: Model, View, and Controller.

Q: What is the role of the controller in Razor Pages?

A: The controller role is taken by the page handler in Razor Pages.

Q: What does the Model component represent in MVC?

A: The data that needs to be displayed, the global state of the application.

Q: What is the View component responsible for in MVC?

A: The template that displays the data provided by the model.

Q: What does the controller do with the model in response to a request?

A: Either fetches the requested data or updates the data that makes up the model.

Q: What does the controller pass to the view?

A: A representation of the model.

Q: What is the view responsible for using the data passed by the controller?

A: To generate the UI.

Q: How does the user initiate an interaction with the controller in web applications?

A: Through an HTTP request.

Q: What contains all the business logic for the application in MVC?

A: The application model.

Q: Is the Razor Page PageModel class considered the model in MVC?

A: No, the Razor Page PageModel class is not the model we're talking about.

Q: What might the controller do when a request to view a product page is received?

A: Contact some product service that's part of the application model.

Q: What does the model do when a product is added to the shopping cart?

A: Update its internal representation of the user's cart.

Q: What is a Razor Page handler compared to in MVC?

A: A mini controller focused on a single page.

Q: What is one of the advantages of the model being independent of the view?

A: It improves testability.

Q: What is the view not involved in, according to the passage?

A: Any of the business logic.

Q: What does the view use to generate the appropriate HTML response?

A: The data passed to it by the controller.

Q: What is a common source of confusion related to MVC?

A: Slightly different uses of the term for slightly different frameworks and types of applications.

Applying the MVC design pattern to Razor Pages

Keywords:

"Razor Pages use this pattern"

"ASP.NET Core also includes a framework called ASP.NET Core MVC"

"controllers and action methods in place of Razor Pages and page handlers"

"Razor Pages builds directly on top of the underlying ASP.NET Core MVC framework"

"avoid Razor Pages entirely and work with the MVC framework directly"

"only option in early versions of ASP.NET Core"

"choosing between Razor Pages and the MVC framework"

"how the MVC design pattern applies to Razor Pages in ASP.NET Core"

"ASP.NET Core implements Razor Page endpoints using a combination of RoutingMiddleware and EndpointMiddleware"

"routing middleware will select which Razor Page handler should be executed"

"endpoint middleware executes the page handler"

"Middleware often handles cross-cutting concerns or narrowly defined requests"

"a more robust framework is required"

"interaction with your application’s core business logic and the generation of a UI"

"mapping the request to an appropriate controller to generating the HTML or API response"

"only a single type of model, which holds all the non-UI data and behavior"

"controller updates this model as appropriate and then passes it to the view"

"vague and ambiguous terms"

"Model, in particular, is such an overloaded term"

"ASP.NET Core uses the word 'model' to describe several related, but different, components"

Questions:

Q: What pattern do Razor Pages use?

A: Razor Pages use this pattern.

Q: What does ASP.NET Core also include besides Razor Pages?

A: ASP.NET Core also includes a framework called ASP.NET Core MVC.

Q: What do controllers and action methods replace in ASP.NET Core MVC?

A: Controllers and action methods replace Razor Pages and page handlers.

Q: What does Razor Pages build directly on top of?

A: Razor Pages builds directly on top of the underlying ASP.NET Core MVC framework.

Q: Can developers choose to avoid Razor Pages?

A: You can avoid Razor Pages entirely and work with the MVC framework directly.

Q: What was the only option in early versions of ASP.NET Core?

A: The MVC framework was the only option in early versions of ASP.NET Core.

Q: What does section 4.2 discuss in greater depth?

A: Choosing between Razor Pages and the MVC framework.

Q: What does the routing middleware do?

A: The routing middleware will select which Razor Page handler should be executed.

Q: What does the endpoint middleware do after routing?

A: The endpoint middleware executes the page handler.

Q: What kind of concerns does middleware often handle?

A: Middleware often handles cross-cutting concerns or narrowly defined requests.

Q: When is a more robust framework required?

A: For requirements that fall outside of these functions, or that have many external dependencies.

Q: What does Razor Pages allow interaction with?

A: Your application’s core business logic and the generation of a UI.

Q: What does Razor Pages handle from mapping to response?

A: Mapping the request to an appropriate controller to generate the HTML or API response.

Q: What does the single type of model in traditional MVC hold?

A: All the non-UI data and behavior.

Q: What does the controller do with the model in traditional MVC?

A: Updates this model as appropriate and then passes it to the view.

Q: What is one problem when discussing MVC?

A: The vague and ambiguous terms that it uses.

Q: Why is the term "model" particularly problematic in MVC?

A: Model, in particular, is such an overloaded term.

Q: What does ASP.NET Core use the word “model” to describe?

A: Several related, but different, components.

DIRECTING A REQUEST TO A RAZOR PAGE AND BUILDING A BINDING MODEL

Keywords:

"routing the request to an appropriate Razor Page handler"

"displaying a list of items that have a given category label"

"make a request to the /category/Simple path"

"maps it against a pre registered list of patterns"

"match a path to a single Razor Page and page handler"

"Razor Page to refer to the combination of the Razor view and the PageModel"

"PageModel class is not the 'model' we're referring to when describing the MVC pattern"

"binding model (if applicable) is generated"

"based on the incoming request, the properties of the PageModel marked for binding, and the method parameters"

"binding model is normally one or more standard C# objects"

"binding model is one or more objects that act as a 'container' for the data provided in a request"

"binding model is a simple string, category, which is bound to the 'Simple' value"

"value is provided in the request URL's path"

"more complex binding model could also have been used"

"corresponds to the method parameter of the OnGet page handler"

"instance of the Razor Page is created using its constructor"

"binding model is passed to the page handler when it executes"

"page handler uses it to decide which todo items to display on the page"

Questions:

Q: What is the first step when your app receives a request?

A: Routing the request to an appropriate Razor Page handler.

Q: What path would be used to request items in the "Simple" category?

A: /category/Simple.

Q: What does routing map the path against?

A: A preregistered list of patterns.

Q: What do routing patterns match to?

A: A single Razor Page and page handler.

Q: What does the term Razor Page refer to in this context?

A: The combination of the Razor view and the PageModel.

Q: What is the PageModel class not considered in the MVC pattern?

A: It is not the “model” we’re referring to when describing the MVC pattern.

Q: When is a binding model generated?

A: Once a page handler is selected.

Q: What is a binding model built based on?

A: The incoming request, the properties of the PageModel marked for binding, and the method parameters required by the page handler.

Q: What are binding models normally composed of?

A: One or more standard C# objects.

Q: What does a binding model act as?

A: A “container” for the data provided in a request.

Q: What is the binding model in the given example?

A: A simple string, category, which is bound to the "Simple" value.

Q: Where is the "Simple" value provided?

A: In the request URL’s path.

Q: What could a more complex binding model contain?

A: Multiple properties populated.

Q: What does the binding model correspond to in the page handler?

A: The method parameter of the OnGet page handler.

Q: How is an instance of the Razor Page created?

A: Using its constructor.

Q: When is the binding model passed to the page handler?

A: When it executes.

Q: What does the page handler use the binding model for in the example?

A: To decide which todo items to display on the page.

EXECUTING A HANDLER USING THE APPLICATION MODEL

Keywords:

"the page handler as the controller in the MVC pattern"

"coordinate the generation of a response to the request it’s handling"

"Validate that the data contained in the binding model provided is valid for the request"

"Invoke the appropriate actions on the application model using services"

"Select an appropriate response to generate based on the response from the application model"

"the application model is a somewhat abstract concept that encapsulates the remaining non-UI parts of your application"

"contains the domain model, a number of services, and the database interaction"

"domain model encapsulates complex business logic in a series of classes that don’t depend on any infrastructure and can be easily tested"

"page handler typically calls into a single point in the application model"

"use a variety of services to check whether the current user is allowed to view certain items"

"to search for items in the given category"

"to load the details from the database"

"to load a picture associated with an item from a file"

"application model will return the required details to the page handler"

"up to the page handler to choose a response to generate"

Questions:

Q: What role does the page handler play in the MVC pattern?

A: The page handler acts as the controller in the MVC pattern.

Q: What is the page handler responsible for coordinating?

A: The generation of a response to the request it’s handling.

Q: What is the first action the page handler should perform?

A: Validate that the data contained in the binding model provided is valid for the request.

Q: What does the page handler use to invoke appropriate actions?

A: The application model using services.

Q: On what basis does the page handler select a response to generate?

A: Based on the response from the application model.

Q: What does the application model encapsulate?

A: The remaining non-UI parts of your application.

Q: What components are included in the application model?

A: The domain model, a number of services, and the database interaction.

Q: What does the domain model encapsulate?

A: Complex business logic in a series of classes that don't depend on any infrastructure and can be easily tested.

Q: Where does the page handler typically call in the application model?

A: Into a single point in the application model.

Q: What might the application model use services for in the to-do list example?

A: To check whether the current user is allowed to view certain items.

Q: What is one action the application model might perform besides authorization?

A: To search for items in the given category.

Q: What database-related action might the application model perform?

A: To load the details from the database.

Q: What file-related action might the application model perform?

A: To load a picture associated with an item from a file.

Q: What happens after the application model returns the required details?

A: It's up to the page handler to choose a response to generate.

BUILDING HTML USING THE VIEW MODEL

Keywords:

"generate a response"

"a view model captures the details necessary for the view to generate a response"

"a view model in the MVC pattern is all the data required by the view to render a UI"

"some transformation of the data contained in the application model"

"extra information required to render the page, such as the page's title"

"used extensively in ASP.NET Core MVC"

"passed to the Razor view to render"

"Razor view can access the Razor Page's page model class directly"

"PageModel typically acts as the view model in Razor Pages"

"Razor Pages use the PageModel class itself as the view model"

"data required by the Razor view exposed via properties"

"Razor view uses the data exposed in the page model to generate the final HTML response"

"sent back through the middleware pipeline and out to the user's browser"

"page handler selects whether to execute the view and the data to use"

"view itself that decides what the content of the response will be"

Questions:

Q: What does a view model capture in the MVC pattern?

A: All the data required by the view to render a UI.

Q: What is the view model typically a transformation of?

A: The data contained in the application model.

Q: What additional information might a view model include?

A: Extra information required to render the page, such as the page's title.

Q: In ASP.NET Core MVC, what is typically passed to the Razor view to render?

A: A single object.

Q: What can the Razor view access directly in Razor Pages?

A: The Razor Page's page model class.

Q: What typically acts as the view model in Razor Pages?

A: The PageModel.

Q: How does the Razor view use the PageModel?

A: By accessing the data required by the Razor view exposed via properties.

Q: What does the Razor view use to generate the final HTML response?

A: The data exposed in the page model.

Q: What happens to the final HTML response?

A: It is sent back through the middleware pipeline and out to the user's browser.

Q: What does the page handler select?

A: Whether to execute the view and the data to use.

Q: Who controls what HTML is generated in the response?

A: The view itself.

PUTTING IT ALL TOGETHER: A COMPLETE RAZOR PAGE REQUEST

Keywords:

"handling a request in ASP.NET Core using Razor Pages"

"steps combine to handle the request to display the list of to-do items for the 'Simple' category"

"traditional MVC pattern is still visible in Razor Pages"

"page handler (controller), the view, and the application model"

"separation of concerns"

"the view is responsible only for taking some data and generating HTML"

"the application model is responsible only for executing the required business logic"

"the page handler (controller) is responsible only for validating the incoming request and selecting which response is required"

"clearly defined boundaries"

"easier to update and test each of the components without depending on any of the others"

"Razor Pages functionality"

"filter pipeline"

"binding models in greater depth in chapter 6"

"generating machine-readable responses using Web API controllers"

"add Razor Pages to your application"

"templates in Visual Studio and the .NET CLI will include Razor Pages by default"

Questions:

Q: What does figure 4.6 show?

A: How the steps combine to handle the request to display the list of to-do items for the “Simple” category.

Q: What components of the traditional MVC pattern are visible in Razor Pages?

A: The page handler (controller), the view, and the application model.

Q: What is the key benefit of the described request-handling process?

A: The separation of concerns.

Q: What is the view responsible for?

A: Taking some data and generating HTML.

Q: What is the application model responsible for?

A: Executing the required business logic.

Q: What is the page handler responsible for?

A: Validating the incoming request and selecting which response is required.

Q: What does having clearly defined boundaries help with?

A: Updating and testing each of the components without depending on any of the others.

Q: What happens if your UI logic changes?

A: You won't necessarily have to modify any of your business logic classes.

Q: What additional Razor Pages feature will be covered in chapter 13?

A: The filter pipeline.

Q: What topic will be discussed in greater depth in chapter 6?

A: Binding models.

Q: How does the MVC design pattern apply when using Web API controllers?

A: The process is identical, apart from the final result generated.

Q: What will the next section show?

A: How to add Razor Pages to your application.

Q: What do some templates in Visual Studio and the .NET CLI include by default?

A: Razor Pages.

Adding Razor Pages to your application

Keywords:

"MVC infrastructure"

"add Razor Pages to an existing project"

"convert an ASP.NET Core application to use Razor Pages"

"pluggable nature of ASP.NET Core"

"ASP.NET Core Web Application"

"ASP.NET Core Empty project template"

"dotnet new web command"

"services.AddRazorPages()"

"MapRazorPages() extension method"

"Pages"

"Add > Razor Page"

"Index.cshtml"

"restore, build, and run your application"

"dotnet run at the command line"

"OnGet handler on the IndexModel"

"Razor Pages rely on a number of internal services"

"AddRazorPages in the ConfigureServices method"

"MapRazorPages in Configure"

"routes that are used to map URL paths to specific Razor Page handlers"

"Razor Pages builds on top of the ASP.NET Core MVC framework"

Questions:

Q: What does the MVC infrastructure serve as in most ASP.NET Core applications?

A: A foundational aspect of all but the simplest ASP.NET Core applications.

Q: What will the result of adding Razor Pages from scratch initially display?

A: “Hello World” on a web page.

Q: What does the example of adding Razor Pages emphasize about ASP.NET Core?

A: The pluggable nature of ASP.NET Core.

Q: Which template should you choose in Visual Studio to create a project without MVC or Razor Pages?

A: ASP.NET Core Empty project template.

Q: What command can you use to create a similar empty project using the .NET CLI?

A: dotnet new web command.

Q: What method is used in ConfigureServices to add Razor Pages?

A: services.AddRazorPages();

Q: What method replaces the basic endpoint in the middleware pipeline?

A: MapRazorPages() extension method.

Q: What folder must be added to the root of the project to contain Razor Pages?

A: “Pages”

Q: How do you create a new Razor Page from the Pages folder?

A: Right-click the folder and choose Add > Razor Page.

Q: What name should you give the Razor Page when prompted?

A: Index.cshtml.

Q: How can you run the project after setup is complete?

A: By pressing F5 from within Visual Studio or by calling dotnet run at the command line.

Q: Which handler is invoked when a request is made to the root "/" path?

A: The OnGet handler on the IndexModel.

Q: What does the OnGet handler cause the Razor Page to do?

A: Render the associated Razor view and send it to the user’s browser.

Q: What will happen if AddRazorPages is not called in ConfigureServices?

A: You’ll get exceptions when your app starts.

Q: What does MapRazorPages do in the Configure method?

A: Registers the Razor Page endpoints with the endpoint middleware.

Q: What happens as part of the MapRazorPages call?

A: Routes that are used to map URL paths to specific Razor Page handlers are registered automatically.

Q: What does Razor Pages build on top of?

A: The ASP.NET Core MVC framework.

Razor Pages vs. MVC in ASP.NET Core

Keywords:

"Razor Pages"

"recommended approach for building server-side rendered applications"

"ASP.NET Core MVC framework behind the scenes"

"use the MVC framework directly"

"creating a Web API"

"use Razor Pages for server-side rendered applications"

"use the MVC framework for building Web APIs"

"misconceptions about Razor Pages"

"equating them to Web Forms"

"underlying basis of the MVC framework"

"similarities and differences between MVC and Razor Pages"

Questions:

Q: What is the recommended approach for building server-side rendered applications with ASP.NET Core?

A: Razor Pages.

Q: What framework does Razor Pages use behind the scenes?

A: The ASP.NET Core MVC framework.

Q: When will you almost certainly use the MVC framework directly?

A: When creating a Web API for working with mobile or client-side apps.

Q: According to the passage, when should you use Razor Pages?

A: For server-side rendered applications.

Q: According to the passage, when should you use the MVC framework?

A: For building Web APIs.

Q: What misconception do some developers have about Razor Pages?

A: Incorrectly equating them to Web Forms.

Q: What is the underlying basis of Razor Pages?

A: The MVC framework.

Q: Why is it useful to understand the similarities and differences between MVC and Razor Pages?

A: Because you'll likely find a use for MVC at some point, even if you use Razor Pages most of the time.

MVC controllers in ASP.NET Core

Keywords:

"ASP.NET Core MVC framework"

"Razor Pages and MVC"

"PageModel and page handler"

"controllers and action methods"

"explicit view models to pass data to a Razor view"

"exposing the data as properties on itself"

"An action (or action method) is a method that runs in response to a request"

"An MVC controller is a class that contains a number of logically grouped action methods"

"MVC controller that provides the same functionality as the Razor Page"

"controller in this case is called ToDo-Controller"

"additional action methods for working with to-do items"

Questions:

Q: What does MVC use instead of a PageModel and page handler?

A: Controllers and action methods.

Q: How do MVC controllers pass data to a Razor view compared to Razor Pages?

A: MVC controllers use explicit view models to pass data, while Razor Pages expose the data as properties on themselves.

Q: What is an action method in MVC?

A: A method that runs in response to a request.

Q: What does an MVC controller contain?

A: A number of logically grouped action methods.

Q: What is the name of the example controller that provides the same functionality as the Razor Page?

A: ToDo-Controller.

Q: What kind of additional action methods might the ToDo-Controller contain?

A: Actions to view a specific item or to create a new one.

An MVC controller for viewing all to-do items in a given category

Keywords:

"ToDoController looks very similar to the Razor Page equivalent"

"architecturally, Razor Pages and MVC are essentially equivalent"

"they both use the MVC design pattern"

"most obvious differences relate to where the files are placed in your project"

Questions:

Q: How does the ToDoController compare to the Razor Page equivalent?

A: The ToDoController looks very similar to the Razor Page equivalent aside from some naming differences.

Q: Architecturally, how are Razor Pages and MVC described?

A: Razor Pages and MVC are essentially equivalent.

Q: What design pattern do both Razor Pages and MVC use?

A: The MVC design pattern.

Q: What is noted as the most obvious difference between Razor Pages and MVC?

A: Where the files are placed in your project.

The benefits of Razor Pages

Keywords:

"code for an MVC controller looks very similar to the code for a Razor Page PageModel"

"single controller can have multiple action methods"

"controllers become very large and bloated, with many dependencies"

"MVC approach traditionally groups classes by type (controller, view, view model)"

"Razor Page approach groups by function"

"Razor Pages embraces the fact that you're building a page-based application"

"Page handlers are functionally equivalent to MVC controller action methods"

"Razor Pages doesn't lose any of the separation-of-concerns that MVC has"

"benefits of using Razor Pages are particularly noticeable when you have 'content' websites"

"Razor Pages is especially optimized for this scenario"

"possible to use both MVC controllers and Razor Pages in the same application"

Questions:

Q: What is a common issue with MVC controllers as described in the passage?

A: Controllers can become very large and bloated, with many dependencies.

Q: How does the MVC approach organize project files compared to Razor Pages?

A: MVC groups classes by type (controller, view, view model), while Razor Pages groups by function.

Q: What is the main benefit of Razor Pages' file layout?

A: It keeps everything related to a single page together, optimizing workflow.

Q: How are Razor Page handlers related to MVC controller action methods?

A: Page handlers are functionally equivalent to MVC controller action methods.

Q: Does Razor Pages maintain separation of concerns compared to MVC?

A: Yes, Razor Pages doesn't lose any of the separation-of-concerns that MVC has.

Q: For what kind of websites are the benefits of using Razor Pages particularly noticeable?

A: For "content" websites, such as marketing websites, where mostly static data is displayed.

Q: Can MVC controllers and Razor Pages be used together in the same application?

A: Yes, it's possible to use both MVC controllers and Razor Pages in the same application.

Ascension campaigns

Keywords:

"When to choose MVC controllers over Razor Pages"

"Razor Pages are great for building page-based server-side rendered applications"

"you should use MVC controllers instead"

"If you already have an ASP.NET application that uses MVC, it’s probably not worth converting your existing MVC controllers to Razor Pages"

"look at doing new development in the application with Razor Pages"

"doing a lot of partial page updates"

"easier to achieve with MVC controllers than Razor Pages"

"all the Razor Pages we’ve looked at have used a single page handler"

"how to define them, how to invoke them, and how to use them to render Razor views"

Questions:

Q: When should you choose MVC controllers instead of Razor Pages?

A: When you don’t want to render views, such as when building a Web API.

Q: What is recommended if you already have an existing MVC application using ASP.NET?

A: It’s probably not worth converting your existing MVC controllers to Razor Pages; keep your existing code.

Q: Which scenario might be easier to achieve with MVC controllers compared to Razor Pages?

A: Doing a lot of partial page updates with JavaScript to avoid full page navigations.

Q: What have all the Razor Pages looked at so far used?

A: A single page handler.

Q: What will the next section discuss about page handlers?

A: How to define them, how to invoke them, and how to use them to render Razor views.

Razor Pages and page handlers

Keywords:

"MVC design pattern"

"controller receives a request and is the entry point for UI generation"

"page handler that resides in a Razor Page's PageModel"

"a page handler is a method that runs in response to a request"

"path of a Razor Page on disk controls the URL path"

"Razor Pages can contain any number of page handlers, but only one runs in response to a given request"

"process of selecting a Razor Page and handler, called routing"

"Confirm that the incoming request is valid"

"Invoke the appropriate business logic corresponding to the incoming request"

"Choose the appropriate kind of response to return"

"PageResult object"

"RedirectToPageResult"

"page handler doesn't generate a response directly; it selects the type of response and prepares the data for it"

"the view generates the response, not the controller"

"page handler is responsible for choosing what sort of response to send"

"page handlers should generally not be performing business logic directly"

"call appropriate services in the application model to handle requests"

"separating concerns ensures your code stays testable and maintainable"

Questions:

Q: What is the entry point for UI generation in the MVC design pattern?

A: The controller receives a request and is the entry point for UI generation.

Q: Where does the entry point lie for Razor Pages?

A: The entry point is a page handler that resides in a Razor Page's PageModel.

Q: What controls the URL path that a Razor Page responds to by default?

A: The path of a Razor Page on disk controls the URL path that the Razor Page responds to.

Q: How many page handlers run in response to a given request?

A: Only one page handler runs in response to a given request.

Q: What are the three general responsibilities of a page handler?

A: Confirm that the incoming request is valid, invoke the appropriate business logic, and choose the appropriate kind of response to return.

Q: What does returning a PageResult object cause?

A: It causes the associated Razor view to generate an HTML response.

Q: Does a page handler generate the response directly?

A: No, it selects the type of response and prepares the data for it.

Q: Who generates the response in keeping with the MVC design pattern?

A: The view generates the response, not the controller.

Q: Should page handlers perform business logic directly?

A: No, page handlers should call appropriate services in the application model to handle requests.

Q: Why is separating concerns important in the context of page handlers?

A: It ensures your code stays testable and maintainable as it grows.

Accepting parameters to page handlers

Keywords:

"requests made to page handlers will require additional values with details about the request"

"request might contain details of the search term and the page number"

"request is posting a form to your application"

"values must be contained in the request"

"no values, such as when a user requests the home page"

"request may contain additional values from a variety of different sources"

"part of the URL, the query string, headers, or the body of the request itself"

"middleware will extract values from each of these sources and convert them into .NET types"

"process of extracting values from a request and converting them to .NET types is called model binding"

"ASP.NET Core can bind two different targets in Razor Pages"

"Method arguments"

"Properties marked with a \[BindProperty] attribute"

"\[BindProperty] attribute does nothing for GET requests"

"Model-bound values can be simple types, such as strings and integers"

"Model-bound values can be a complex type"

"values provided in the request are not bound to a property or page handler argument, the additional values will go unused"

Questions:

Q: What kind of additional values might a request to a search page contain?

A: Details of the search term and the page number they're looking at.

Q: What sources can additional values in a request come from?

A: They could be part of the URL, the query string, headers, or the body of the request itself.

Q: What is the process of extracting values from a request and converting them to .NET types called?

A: Model binding.

Q: What two different targets can ASP.NET Core bind in Razor Pages?

A: Method arguments and properties marked with a \[BindProperty] attribute.

Q: What does the \[BindProperty] attribute do by default for GET requests?

A: It does nothing for GET requests by default.

Q: What happens to values in the request that are not bound to a property or page handler argument?

A: The additional values will go unused.

Returning responses with ActionResults

Keywords:

"page handlers decide what type of response to return, but they don’t generate the response themselves"

"IActionResult returned by a page handler"

"Razor Pages infrastructure using the view engine will generate the response"

"following the MVC design pattern"

"separates the decision of what sort of response to send from the generation of the response"

"test your action method logic to confirm that the right sort of response is sent"

"separately test that a given IActionResult generates the expected HTML"

"ASP.NET Core has many different types of IActionResult"

"PageResult—Generates an HTML view for an associated page in Razor Pages"

"ViewResult—Generates an HTML view for a given Razor view when using MVC controllers"

"RedirectToPageResult—Sends a 302 HTTP redirect response to automatically send a user to another page"

"RedirectResult—Sends a 302 HTTP redirect response to automatically send a user to a specified URL"

"FileResult—Returns a file as the response"

"ContentResult—Returns a provided string as the response"

"StatusCodeResult—Sends a raw HTTP status code as the response"

"NotFoundResult—Sends a raw 404 HTTP status code as the response"

"executed by Razor Pages, will generate a response to send back through the middleware pipeline"

"you generally won’t use some of these action results, such as ContentResult and StatusCodeResult"

"you will likely use them if you are building Web APIs with MVC controllers"

Questions:

Q: What does a page handler decide and what generates the response in Razor Pages?

A: Page handlers decide what type of response to return, but the IActionResult returned by a page handler generates the response.

Q: How does following the MVC design pattern help in testing?

A: It separates the decision of what sort of response to send from the generation of the response, allowing you to test action method logic and separately test the IActionResult.

Q: What does a PageResult do in Razor Pages?

A: It generates an HTML view for an associated page in Razor Pages.

Q: What is the difference between RedirectToPageResult and RedirectResult?

A: RedirectToPageResult sends a 302 HTTP redirect to another page, while RedirectResult sends a 302 HTTP redirect to a specified URL.

Q: Which IActionResult types might not commonly be used in Razor Pages but are useful in Web APIs with MVC controllers?

A: ContentResult and StatusCodeResult.

Q: What happens when an IActionResult is executed by Razor Pages?

A: It generates a response to send back through the middleware pipeline and out to the user.

PAGERESULT AND REDIRECT TO PAGE RESULT

Keywords:

"building a traditional web application with Razor Pages"

"PageResult, which generates an HTML response using Razor"

"various redirect-based results to send the user to a new web page"

"web application sends HTTP redirects whenever it needs you to move to a different page"

"browser automatically follows the redirect requests"

"whenever you return HTML you use a PageResult"

"when you redirect to a new page, you use a RedirectToPageResult"

"Razor Pages are generally designed to be stateless"

"persist data between multiple pages, you need to place it in a database or similar store"

"TempData, which stores small amounts of data in cookies for a single request"

Questions:

Q: What does PageResult generate in a traditional web application with Razor Pages?

A: PageResult generates an HTML response using Razor.

Q: When does a web application send HTTP redirects?

A: The web application sends HTTP redirects whenever it needs you to move to a different page, such as when a user submits a form.

Q: What result type do you use when redirecting to a new page in Razor Pages?

A: You use a RedirectToPageResult when redirecting to a new page.

Q: How are Razor Pages generally designed in terms of state?

A: Razor Pages are generally designed to be stateless.

Q: Where should you store data to persist it between multiple pages in Razor Pages?

A: You need to place it in a database or similar store.

Q: What is TempData used for in Razor Pages?

A: TempData stores small amounts of data in cookies for a single request.

NOT FOUND RESULT AND STATUS CODE RESULT

Keywords:

"send specific HTTP status codes"

"404 HTTP status code is returned to the browser"

"NotFoundResult, which will return a raw 404 HTTP status code"

"StatusCodeResult and setting the status code returned explicitly to 404"

"NotFoundResult doesn’t generate any HTML"

"StatusCodePagesMiddleware to intercept this raw 404 status code"

"provide a user-friendly HTML response for it"

Questions:

Q: What HTTP status code is returned when a requested product does not exist in an e-commerce application?

A: A 404 HTTP status code is returned to the browser.

Q: How can Razor Pages return a raw 404 HTTP status code?

A: Razor Pages can return a raw 404 HTTP status code by returning a NotFoundResult.

Q: What is the alternative to NotFoundResult for returning a 404 status code?

A: You can use StatusCodeResult and set the status code explicitly to 404.

Q: Does NotFoundResult generate any HTML?

A: No, NotFoundResult doesn’t generate any HTML; it only returns a raw 404 status code.

Q: How can a user-friendly HTML response be provided for a raw 404 status code?

A: The StatusCodePagesMiddleware can intercept the raw 404 status code and provide a user-friendly HTML response.

CREATING ACTIONRESULT CLASSES USING HELPER METHODS

Keywords:

"ActionResult classes can be created and returned using the normal new syntax of C#"

"Page method to generate an appropriate PageResult"

"RedirectToPage method to generate a RedirectToPageResult"

"NotFound method to generate a NotFoundResult"

"returning an IActionResult doesn’t immediately generate the response"

"execution of an IActionResult by the Razor Pages infrastructure"

"response passes back through the middleware pipeline"

"MVC and Razor Pages infrastructure in ASP.NET Core runs as part of the EndpointMiddleware pipeline"

"RoutingMiddleware decides which Razor Page and handler to invoke for a given request"

"role of the routing infrastructure"

"define routes, how to add constraints to your routes, and how they deconstruct URLs"

Questions:

Q: How can ActionResult classes be created and returned in C# according to the passage?

A: They can be created and returned using the normal new syntax of C#.

Q: What helper methods does the Razor Pages PageModel base class provide for generating responses?

A: It provides methods like Page to generate a PageResult, RedirectToPage to generate a RedirectToPageResult, and NotFound to generate a NotFoundResult.

Q: When is the response generated after returning an IActionResult in Razor Pages?

A: The response is generated when the IActionResult is executed by the Razor Pages infrastructure, which occurs outside the action method.

Q: Through what pipeline does a response pass before being sent to the user?

A: The response passes back through the middleware pipeline before the ASP.NET Core web server sends it to the user.

Q: What middleware decides which Razor Page and handler to invoke for a given request?

A: The RoutingMiddleware decides which Razor Page and handler to invoke.

Q: Why is it impractical to have a Razor Page for every URL in an app?

A: Because, for example, every product in an e-shop would need its own Razor Page, making it difficult to manage.

Q: What will be covered in the next chapter related to routing?

A: How to define routes, add constraints to routes, and how they deconstruct URLs to match a single Razor Page handler.